Assignment 6: Evaluating your Classifier

Overview

The purpose of this assignment is to apply what you've learned about Supervised Machine Learning performance metrics. You will build a k-fold cross validation system, calculating accuracy and class specific precision, recall and f1 measure.

More Details

Please use the same groups that you used for Assignment 5.

K-fold cross validation enables us to optimize our models while avoiding overfitting, that is, training systems that perform well on training data but do not generalize to unseen data.

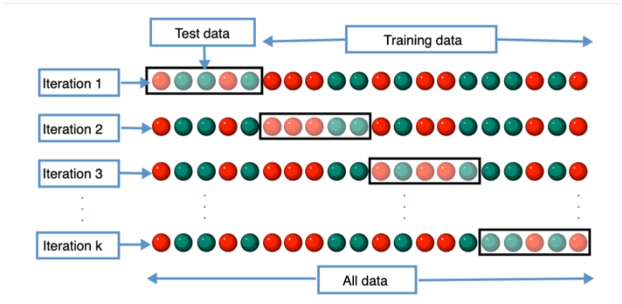

Figure 1: K-fold Cross Validation visual - from Wikipedia

In k-fold cross validation, we evenly (and randomly) divide our data into k sets. We run k iterations of training and testing, allowing each of the k sets a turn to be the testing set, training with the remainder of the data. On each iteration, when classifying the testing data, we keep track of the count of true positives (tp), true negatives (tn), false positives (fp) and false negatives (fn).

These counts then enable us to calculate our performance metrics as follows…..

$$Accuracy = \frac{tp+tn}{tp+fp+tn+fn} \qquad Precision = \frac{tp}{tp+fp} \qquad Recall = \frac{tp}{tp+fn} \qquad f1 = \frac{2*precision*recall}{precision+recall}$$

Accuracy is an overall performance measure, while *precision*, *recall* and *f1* are specific to the positive class. To calculate these 3 values from the negative class perspective, we simply need to swap the positives and negative (e.g. *precision* = tn/(tn + fn)).

After calculating these 7 performance metrics for each of the k iterations, we need to summarize them into one set of 7 scores. This set of 7 scores will then enable us to reason about the performance of our system. If we add more features, we can run the k-fold cross validation and directly compare the system(s) performance. This enables us to optimize our system without overfitting to the training data.

Your Job
Notice that the evaluate method has already been implemented in *Assignment_6.py*. This method drives the k-fold cross validation but calls 4 methods that have not yet been implemented. Your job is to implement these 4 methods according to the provided doc strings.

- *split*
- *classify_all*
- *analyze_results*
- *calculate_averages*

Your code should pass the relevant tests before moving on to the next method.

For *split*, to randomly divide the corpus, you'll need to use the shuffle function from random. Here's an example use:

```
import random
x = ['a', 'b', 'c', 'd', 'e']
random.shuffle(x)
print(x)
#prints this…['d', 'a', 'e', 'b', 'c']
random.shuffle(x)
print(x)
#prints this…['b', 'a', 'd', 'c', 'e']
```

(Optional) Extensions
Notice the output of evaluate method. Save the 7 scores. Mine look like this:

```
summary of results
average 0.8081367615275823
positive precision 0.9758507055614508
positive recall 0.7803049474167296
positive f-measure 0.8671352271158523
negative precision 0.5079256773115968
negative recall 0.9213844549610973
negative f-measure 0.6547272970003375
```

Feature selection is one of the most important factors in supervised ML system performance. Thus far, you've only used "unigrams" (i.e. single words) as features. Another common feature is "bigrams" (i.e. two word phrases). Another feature might be the length of the document, or amount of capitalization or punctuation used. You could also remove frequently occurring words (stop_words) that are often thought of as noise. See `sorted_stoplist.txt` (provided) or store the frequencies of the stems of the words instead of the words themselves (try porters stemmer).

Consider writing a different set of train and classify functions that implement/utilize a different feature set. Normally when we implement another feature set, we combine it with unigrams. That is, unigrams are almost always useful.

If you implement another version of the classify function, it would be useful to see if the new feature set improves or diminishes system performance. Run k-fold cross validation again, now using your new train/classify. How do your scores compare?